高等教育におけるノーコード開発プラットフォームの有用性
ーAppSheet のプラットフォームで作成した単語学習アプリを中心にー

カーク・マスデン

熊本学園大学

masden@kumagaku.ac.jp

The Utility of No-Code Developent Platforms in Higher Education: A Report on a Vocabulary App Built on the AppSheet Platform

Kirk Masden

Kumamoto Gakuen University

概要

本稿では AppSheet というノーコード開発プラットフォームで作成した単語学習アプリについて報告する。このアプリは、認知心理学や教育学において注目されてきた間隔反復、検索練習、精緻化などの学習戦略を単語学習に効率よく活かす目標で開発した。ここでは、語学教育や学習理論の観点からの考察は最小限にし、筆者が開発したアプリを一つの事例として紹介し、大学教育におけるノーコード開発プラットフォームの有用性を中心にして考察する。

1　はじめに

AppSheetとは、Googleスプレッドシートなどのクラウド型の表計算サービスと連携したノーコード開発プラットフォームである。筆者がAppSheetを利用するようになったのは、2016年に作成した単語学習用のGoogle スプレッドシートがきっかけだった。パソコン上では、そのスプレッドシートには一定の利用価値があったものの、スマホでは操作性の悪さが課題となっていた。そこで、スマホでの操作性を改善する方法をウェブ上で探していたところ、AppSheetの存在を知ることとなり、同年の秋にそのプラットフォームでアプリの最初の試作版を作った。

AppSheetの開発プラットフォームはその二年前の 2014 年に、Praveen Seshadri 氏により創設された[1]。筆者が利用し始めた2016年には会社はまだ比較的小さく、掲示板での筆者からの問い合わせに対して、創設者兼CEOのSeshadri氏自らが答えることが多々あった。2020 年には AppSheet が Google 社に買収され[2]、現在はノーコード開発で屈指のプラットフォームとなっている[3]。

AppSheetは主にビジネスにおける利用を想定したプラットフォームで、ユーザーが作ったアプリを会社等の組織の中で使用することは有料となる。しかし、アプリを開発したり、「公開サンプル」として発表することは無料だ。筆者のアプリはこうした「サンプル」としてAppSheetのサイト内で公開されたもの[4]で、一人一人の受講生は独立したユーザーとしてアプリをコピーするだけなので、授業での利用に費用はかからない。

2　アプリの仕組み

筆者が作成したアプリは英語で言う flashcard app の一種である。間隔反復（spaced repetition）の復習スケジュールで勉強を進めていくことを主な目的に作成したもので、その目的にちなんで“Kankaku”と呼んでいる。同様のアプリはたくさんあるが、有料であったり、筆者が求める機能がないなどの問題があり、自ら作ることにした。

単語学習にアプリを利用する学生は、図1に見られるような問いを見てから答えを見る。正解できたら、“GOT IT” のアイコンをタップし、できなかったら“NEXT”をタップする。“NEXT”は不正解を意味するので、補習としてスケジュール内のそのカードの復習回数が増やされる。

勉強するカードについては、教員が用意したものを学生が利用するだけでなく、学習者自身も作成できる。カードを作成する際には、文字データの他に、画像も追加できる。画像はユーザーの Google

ドライブ内のフォルダにアップロードされ、スプレッドシートにはアプリ内で入力した文字データと画像へのリンクが書き込まれる。

図 1 アプリ内の問いと答えの例

なお、AppSheet アプリ内で画像は追加できるが、利用者が録音・録画したファイルはアプリ内でアップロードすることはできず、再生のみとなっている。例えば、図 1 の中の答えの画面に"Play"というアイコンがあるが、これをタップすると、大学のサーバーに載せた mp3 が端末のブラウザーで再生される。動画については YouTube の動画または gif のアニメーションファイルをアプリ内で再生できるが、mp4 は mp3 と同様に端末のブラウザーで再生するためのリンクを貼ることになる。

学習者がアプリを利用していると、それぞれのカードについて次のデータがスプレットシートに記録される。

- 作成日付と時刻
- 勉強開始の日付と時刻
- 閲覧回数
- 正解を意味する GOT IT をタップした回数や日付と時刻

また、各々のカードに関する統計とは別に「セッション」という勉強の単位に関する数値や日付と時刻もスプレッドシートに記録される。

これらの基礎データを使って、アプリの利用頻度、使用する際の速度、正解率など、様々な統計が自動的に算出され、図 2 で見られるような形で、アプリの中で表示される。また、学習者がスプレッドシートを教員と共有していれば、教員も同じ統計を確認できる。統計の中には、主に教員にとって指導や研究に役立つと考えられるものもあるが、図 2 の中の「1 日当たりの記録」などは学習者にとって大変有用であると考えている。学習者が「1 日当たりの記録」を普段から確認することを習慣化すれば、一定の学習促進効果が期待できるだろう。

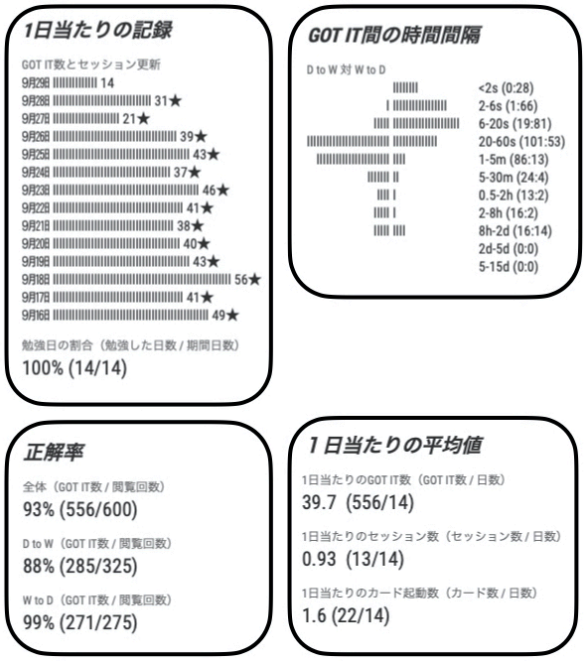


図 2 アプリ内で確認できる統計の一部

3 データの共有

学習者が筆者のアプリを独学で利用するだけなら、Google スプレッドシートは単なるデータの保管場所となる。しかし、学習者がスプレッドシートを教員と共有すると、アプリの教育現場での用途がかなり広がる。

図 3 は、スプレッドシートの共有で可能となる情報の流れを示している。図内のパソコンのイメージはクラウドにあるスプレッドシートを表している。「学生」は一人ひとりの学習者のデータを表している。スマホなどの端末を利用すると、AppSheet のサーバーを介してデータがこれらのスプレッドシートに記録される。「教員」は、教員が管理に利用するスプレッドシートを表している。Google スプレッドシートの importrange という関数を利用すると、全学生の主なデータを集約できる。「公開」は学生へのフィードバック用の公開スプレッドシートを表している。

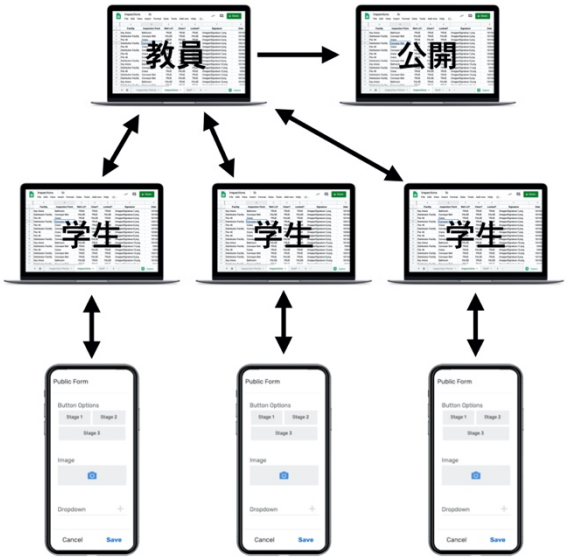


図 3 データ共有の流れ

図4はそのフィードバック様式の一例である。公開のスプレッドシートには氏名や学籍番号などの個人情報はなく、その代わりに学生自らが設定する secret name が表示される。筆者が学生に公開するシートでは、データがランクを示す降順に並べられているため、「番付」と呼んでいる。

	A	B	C	D	E	F
2			過去2週間の記録です。ノルマ達成後に			
3			4以上	4以下	4以上	
4	Secret name	ノルマ判定	勉強日数	最新利用日からの経過日数	1日当たりのGOT IT数	GOT IT数 (D to W + W to
6	Rasu	○	7	1	15.0	120 (60+60)
7	TATA	○	5	1	15.6	109 (55+54)
8	namie88	○	4	2	14.6	102 (51+51)
9	do.8881		3	1	19.4	97 (47+50)
10	yamajan	○	5	2	13.6	95 (53+42)
11	Eagle Eye	○	4	1	11.9	83 (37+46)
12	しろくま1号	○	8	0	9.1	82 (46+36)
13	niko.and.8	○	5	1	10.0	80 (40+40)
14	aiueo	○	5	1	9.3	74 (37+37)
15	hulk	○	4	0	8.8	70 (34+36)

図 4 「番付」のフィードバック例

なお、図 3 で見られる「学生」と「教員」のスプレッドシートとの間に双方向性を示す矢印がある。「学生」から「教員」へのデータの流れは上述したとおりだが、「教員」から「学生」への流れとしてはカード(教材)の提供がある。アプリと一緒にコピーされる Google スプレッドシートの中に、筆者が管理するスプレッドシートのデータを importrange で読み込むシートが含まれており、アプリ内でこのテータを閲覧することができる。importrange のデータは閲覧のみだが、アプリ内でコピーすれば、修正したり勉強のスケジュールに組み込んだりすることができる。

図5では、スプレッドシートの共有による課題提出の一例を示している。この場合、アプリ内で勉強できる単語カードを作成することが課題だった。オレンジ色の列には筆者が指定した課題が表示されており、青の列は学生がアプリを通して提出したデータである。課題をする学生は「番付」（公開シート）の中で示した課題文の URL をコピーし、アプリ内のカード作成用のフォームに貼り付ける。筆者が課題用に指定したタグ (“kadai1”など）を正しく入力できていれば、importrange、regexmatch、filterなどの関数の組み合わせにより、提出用のデータのみが自動的に選び出されるようにしている。さらに、URL が指定したリストにあること、そのURLが他の学生によって先に利用されていないことなど、表計算の機能を使って機械的に確認できる条件をめぐる「自動審査」に合格できたら、筆者が「人為審査」を行う仕組みだ。

4 アプリを使った学習評価と指導

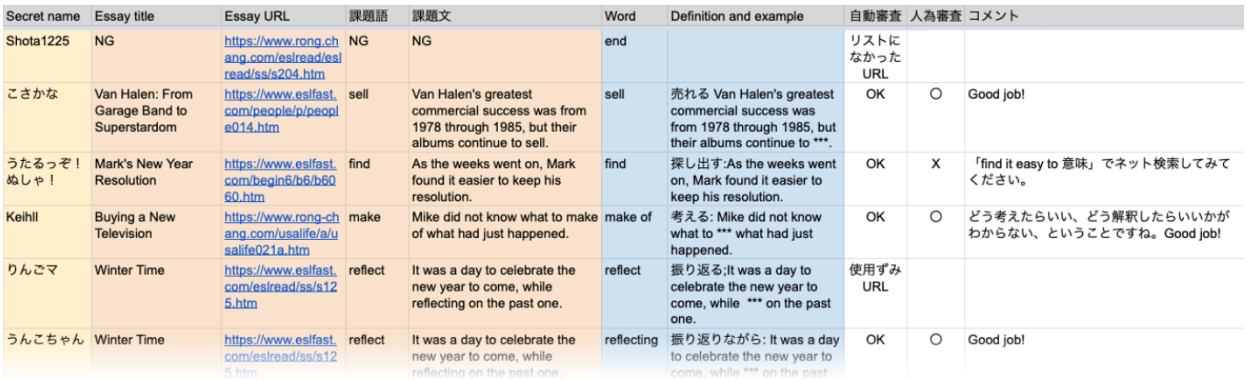
筆者の授業におけるアプリ関連の学習評価内容はノルマ、小テスト、課題の 3 つに分けられる。ノルマを通して、アプリの利用に関する数値が最低基準を上回っているかどうかを確認する。小テストで、アプリで勉強している内容の習熟度を測る。そして、単語カードを作成する課題により、アプリの継続的かつ自律的な活用に必要なスキルを身に付けることを目指している。

4.1 ノルマ

現在、筆者の授業において、ノルマ判定の対象期間を過去 2 週間とし、その期間内に次の 3 種類の数値をめぐるノルマを課している。

- 勉強量（閲覧したカード数および正解数）
- 使用頻度（アプリを使用した日数）
- 使用速度（正解間の時間間隔の平均値）

勉強量および使用頻度をめぐるノルマは、一定以上の量の勉強に継続的に取り組ませるために設けている。使用速度をめぐるノルマは、速度を落として、じっくりと考えさせるために設けている。



Secret name	Essay title	Essay URL	課題語	課題文	Word	Definition and example	自動審査	人為審査	コメント
Shota1225	NG	https://www.rong.chang.com/eslread/eslread/ss/s204.htm	NG	NG	end		リストになかったURL		
こさかな	Van Halen: From Garage Band to Superstardom	https://www.eslfast.com/people/p/people014.htm	sell	Van Halen's greatest commercial success was from 1978 through 1985, but their albums continue to sell.	sell	売れる Van Halen's greatest commercial success was from 1978 through 1985, but their albums continue to ***.	OK	○	Good job!
うたるっぞ！ぬしゃ！	Mark's New Year Resolution	https://www.eslfast.com/begin6/b6/b6060.htm	find	As the weeks went on, Mark found it easier to keep his resolution.	find	探し出す:As the weeks went on, Mark found it easier to keep his resolution.	OK	X	「find it easy to 意味」でネット検索してみてください。
KeihII	Buying a New Television	https://www.rong-chang.com/usalife/a/usalife021a.htm	make	Mike did not know what to make of what had just happened.	make of	考える: Mike did not know what to *** what had just happened.	OK	○	どう考えたらいい、どう解釈したらいいかがわからない、ということですね。Good job!
りんごマ	Winter Time	https://www.eslfast.com/eslread/ss/s125.htm	reflect	It was a day to celebrate the new year to come, while reflecting on the past one.	reflect	振り返る;It was a day to celebrate the new year to come, while *** on the past one.	使用ずみURL		
うんこちゃん	Winter Time	https://www.eslfast.com/eslread/ss/s125.htm	reflect	It was a day to celebrate the new year to come, while reflecting on the past one.	reflecting	振り返りながら: It was a day to celebrate the new year to come, while *** on the past	OK	○	Good job!

図 5「番付」での課題提出の一例

計算方法は、対象期間における正解と正解との間の時間間隔を 5 分未満と 5 分以上とに分類した上で、5 分未満の平均値を求める。多くの利用者は 20 秒以上の平均値になるが、内容について考えずに、形だけアプリを操作する学生の平均値は 3 秒未満になることもある。速度をめぐるノルマを設けることにより、こうした形だけの利用を未然に防ごうとしている。

ノルマ判定の対象期間を過去 2 週間としているのは、継続的な利用を促進するためである。授業において遅れを取った学生でも、これから 2 週間の間に頑張ればノルマを達成できる。また、逆に、それまで精力的に勉強してきた学生でも、一旦アプリを使用しなくなると、2 週間以内にノルマ達成基準を下回ることになる。また、13 日間や 15 日間ではなく、14 日間にしているのは、ただ単に 2 週間というのが利用者にとってわかりやすいからである。ちなみに、ノルマ判定期間を 2 週間としていることに合わせて、アプリ内で表示される統計を短期（過去 2 週間）と長期（過去 5 ヶ月）に分けている。

学生へのノルマ達成に関する通知は図 4 に見られるように、公開のスプレッドシートで行っている。このスプレッドシートが準備できたら、後は学生の利用状況に合わせて数値や判定結果が自動的に変わっていくので、教師にとって負担にならない。また、毎週のノルマ判定時刻を決め、Google script を利用して、全学生のその時刻での判定結果が自動的に記録されるようにしている。こうして記録した複数回のノルマ判定結果を成績評定の一つの基準にしている。

4.2 小テスト

一人ひとりの学生がアプリ内で実際に勉強してきたデータを使って個別の小テストを作成している。指定する基準に合わせて、スプレッドシートは小テスト用のデータを無作為に選び出し、印刷可能な形で表示する。結果的に教師にとっての作業時間は一人につき 30 秒程度で済む。また、採点時に利用できる問いと答えの入った pdf も印刷の過程で作成される。

作成された小テスト内にそれぞれの問いのアプリ内の正解率が表示される。アプリ内での正解率と小テストでの正解率に大きな差がある場合には、学生に対しその問題を指摘し、普段の取り組みの改善に向けた指導ができる。

4.3 課題

筆者の授業では、アプリ用のカード作成を課題にしている。図 5 に見られるように、筆者は複数の URL における文とその文中の単語を提示し、学生はその中から取り組む内容を選ぶ。簡単なようだが、様々な間違いがありうる。学生が指定外の URL や他の学生が先に取り組んでいる URL を選んだ場合、スプレッドシートの関数で「リストになかった URL」や「使用ずみ URL」などのエラーメッセージを自動的に表示する。自動審査を通っても、図 5 の find に関する課題のように、学生が選ぶ訳語や定義が文脈と合わないことがしばしば見受けられる。したがって、カード作成の課題は、

アプリ操作のみならず、文脈における言葉の意味を考える重要性を認識させる上でも効果があることを実感している。

5 考察

以上報告した取り組みについて、考察すべき内容を二種類に大別できる。一つは言語習得や語学教育におけるアプリの有用性をめぐる考察で、もう一つはアプリ作成に利用したノーコード開発プラットフォームの大学教育における可能性をめぐる考察である。本稿では前者を割愛し、後者に絞って考えたい。

5.1 データ共有によるメリット

筆者の授業において、上で説明したデータ共有の仕組みには、次のメリットがある。

- 学生の普段のアプリの利用状況を確認できる。
- 学生の利用状況や一人ひとりの学生が勉強している内容に合わせた小テストを個別に作成できる。
- ウェブを通して、自動的にアップデートされるデータで学生へのフィードバックができる。
- 匿名性を保ちながら、他の学生も閲覧できる形で、個別の課題に対するフィードバックや指導をウェブ上でできる。

また、現時点では実行できていないが、今後、学生のアプリからのデータを勉強法に関する数量的な研究に役立てることも考えられる。

以上のデータ共有によるメリットは語学教育に限らず、様々な教育分野においても有用性が期待できるだろうと思われる。

5.2 ノーコード開発の難易度

そもそも、プログラミングの知識を必要としないノーコード開発プラットフォームおよびノーコード的な要素と最小限のプログラミグを組み合わせるローコード開発ブラットフォームは、アプリ開発の「民主化」を図るために作られてきたと言えよう。言い換えれば、こうしたプラットフォームの主な目標はアプリ開発の難易度を下げることにある。そこで、ノーコードやローコードが日本や欧米のビジネス界で注目されている。欧米の多くのビジネス誌はこうしたプラットフォームに関する記事を掲載しており、最近日本でもノーコードやローコードに関する記事が『日経コンピュータ』に掲載されている[5]。

従来のプログミングと比較して、ノーコードやローコードの開発プラットフォームがアプリ開発を容易にしていることは間違いないが、実際の難易度はどの程度だろうか。筆者は AppSheet 以外の開発ツールを利用したことがないため、比較はできないが、AppSheet を利用した経験から、プラットフォーム学習およびアプリ開発に要する時間や難度についてコメントしたい。

本稿の「はじめに」で述べたように、筆者は2016年からAppSheetを利用したアプリ開発に取り組んでいる。当初は掲示板で指導を受けながら、数週間程度で最初のバージョンを作ることができたと記憶している。アプリを学生に使用させるだけでなく、筆者は自らの日本語の勉強にも利用し、当初から一定の利用価値があると感じた。しかし、その反面、最初のバージョンは機能が少なく、学生の混乱を招く側面があったため、それから 4 年間をかけて機能の拡充やUX（ユーザー・エクスペリエンス）の改善に努めてきた。したがって、ここで報告したアプリ開発に要した時間は、考え方によっては数週間とも、数年間とも言える。

AppSheet のウェブサイトにおける宣伝文句の一つは「数分でのアプリ開発」（applications developed in minutes）である。確かに、簡単なスプレッドシートをスマホで表示させるためのアプリを数分で、ほぼ自動的に作ることが可能だ。しかし、プロジェクトの目的や性質によっては開発にかなりの時間を要する場合がある。その時間を、開発ノウハウの習得に必要な時間と、開発作業そのものに必要な時間に大別できる。

上で述べたように、筆者が開発ノウハウを身につける上で、AppSheet のウェブサイトにある利用者のための掲示板が大変役に立った。開発の途中で行き詰まったときには、その問題について掲示板に投稿し、AppSheetの従業員や他の利用者に助けてもらった。2020 年 10 月 16 日現在までの筆者

の投稿数は 160 件ある。その一部には筆者がわかってきたコツなどを他の利用者のためにまとめたもの含まれているが、投稿の大多数は筆者が直面した困難や抱いた疑問などに関する問い合せである。

中でも、表計算の関数に相当する expression が難しいと感じた。筆者は SQL 関数を利用したことがないが、AppSheet の expression は SQL 関数と似ているようだ。上記の 160 件の投稿の中に、expression に関するものがかなりの比重を占めている。また、アプリが大きくなるにつれ、速度が問題となった。ノーコードでも、使い方次第で計算等にかかる時間が増え、UX が悪化する場合がある。アプリ内の無駄をなくし、UX を改善する方法を学習することに時間を要した。

時間数で示すことはできないが、以上のように、筆者はかなりの時間をこうした学習に費やしたことは間違いない。

プラットフォームの学習に必要な時間を差し引き、アプリ作成作に要した時間に絞ってもかなりの時間を要する場合がある。それはノーコードでも、アプリを構成するパーツが多くなればなるほど、そのパーツ作成や組み立てに時間を要するからである。現時点での筆者のアプリのパーツ数は次のとおりだ。

- 10 Tables
- 675 Columns
- 44 Slices
- 163 Views
- 54 Format Rules
- 237 Actions

ここでは、それぞれの項目の意味や役割に関する解説を差し控えるが、構成パーツは 1000 個を超えているので、その組み立てだけでも時間がかかったことは想像に難くないだろうと思う。

5.3 ノーコードの可能性と将来性

ノーコードは取っつきやすいが、高度なアプリ作成には向いていないとの見方がある。例えば、Qiita というプログラミング情報のナレッジコミュニティで発表された記事では「ノーコードのツールは機能制限があるため広範囲のシステム」には向いていないと評している[6]。また、「ノーコードは開発者を殺すか？」と題したネット記事において、ノーコードには限界があるため、これからも「ノーコードでできない部分を開発するエンジニア」が必要だと結論づけている[7]。

以上のような指摘が誤っているとは思わないが、実際に利用した経験から、ノーコードの限界よりも、その可能性や汎用性を強く感じている。アプリ内の録音など、まだ AppSheet のプラットフォームに必要な機能がないためできないこともあるが、AppSheet のプラットフォームにある機能の組み合わせで、筆者がアプリ内に望んでいた機能のほとんどを追加することができた。また、Google スプレッドシートや Google フォームなどとの組み合わせで、従来のプログラミングだけでは実現が困難なデータの共有などが可能となっていることも注目に値すると思う。

さらに、AppSheet で利用できる機能等の今後の拡充を大いに期待できる。筆者が AppSheet を利用してきた 4 年間でも、その機能は大幅に増えた。例えば、現在、アプリ内でデータをスプレッドシートに書き込んだりするなど、様々な作業を行うために利用できる action は、筆者が AppSheet を利用し始めた 2016 年にはまだなかった。今後更なる補充が期待できる。機能が豊富になるにつれ、従来のプログラミグやローコードとの格差が縮まり、ノーコードの有用性は増すだろう。

6 大学教育における有用性

大学教育におけるノーコード開発プラットフォームの有用性について次のことが言えよう。

- 大学教育でアプリ開発を行う必要が生じた場合に、ノーコード開発プラットフォームを利用することで、従来のプログラミグに比べて、開発にかかる時間や費用の削減が期待できる。
- AppSheet のように、クラウド上でデータ共有を可能とするサービスにデータが保管される場合、そのデータ共有によるメリットは注目に値する。

なお、IT 教育において、将来性の高いノーコード開発プラットフォームの利用法を教えることも検討に値すると思われる。

上述したように、ノーコードやローコード開発プラットフォームはビジネス界においては注目を集めているが、高等教育での活用に関する考察は

まだ本格的には始まっていないようだ。高等教育機関におけるローコードの活用については、2019年に発表されたノルウェー経済大学の修士論文があるが、その参考文献表には高等教育での利用に関する先行研究は見当たらない[8]。日本語で発表されたノーコード開発プラットフォームの高等教育における利用に関する考察としては、本稿が最初と思われる。本稿が更なる議論や研究に向けた小さな一歩となれば、幸甚である。

なお、上述したように、本稿で報告したアプリはネットで公開されており、Google のアカウントさえあれば、AppSheet でアプリをコピーできる。コピー後は、筆者と同様にアプリの中身を確認したり、編集したりすることができる。また、筆者は AppSheet の授業における利用に関する動画もネットで公開している[9]。内容は本稿とほぼ同じだが、アニメーション等を利用しているため、本稿より理解しやすい面があると思う。

アプリそのものをネットで公開しているが、学生のデータを管理するための Google スプレッドシートの公開準備はまだ整っていない。アプリのみならず、そのアプリから得たデータを管理するためのスプレッドシートの雛形と利用法に関する解説を公開することを今後の課題としたい。

参考文献

[1] Praveen Seshadri, AppSheet Launches DIY Platform for Anyone to Create Mobile Apps From Spreadsheets, 2014 年 10 月 22 日, https://blog.appsheet.com/2014/10/22/appsheet-launches-diy-platform-for-anyone-to-create-mobile-apps-from-spreadsheets/

[2] Ron Miller, Google acquires AppSheet to bring no-code development to Google Cloud, TechCrunch, 2020 年 1 月 15 日, https://techcrunch.com/2020/01/14/google-acquires-appsheet-to-bring-no-code-development-to-google-cloud/

[3] Best No-Code Develpment Platforms Software in 2020, G2, https://www.g2.com/categories/no-code-development-platforms

[4] Kirk Masden's portfolio page, https://www.appsheet.com/portfolio/230844

[5] 中島 募、大川原 拓磨、ローコード革命（特集）、日経コンピュータ、2020 年 7 月 23 日号、pp.26-39.

[6] @kento_gm、ノーコード・ローコードのあれこれ、Quita、2020 年 3 月 3 日更新、https://qiita.com/kento_gm/items/34c46a9bc6b3017c9345

[7] 金春 利幸、ノーコードは開発者を殺すか？、R3 Cloud Journey (blog)、2020 年 8 月 12 日、https://www.r3it.com/blog/20200812-does-nocode-kill-developpers

[8] Erik Vikebø & Ludvig B. Sydvold, An Inquiry into Low-Code Solutions in Institutions for Higher Education, Norwegian School of Economics, 2019, https://core.ac.uk/download/pdf/288306736.pdf

[9] Kirk Masden、How I Use AppSheet in My Classes、2020 年 5 月 1 日, http://kguics.blogspot.com/2020/05/how-i-use-appsheet-in-my-classes.html