Adaptive Ogmented: Teaching Inheritance through Augmented Reality

Anto Oswin Nihal
Arizona State University
Tempe AZ
United States
anihal@asu.edu

Denim Datta
Arizona State University
Tempe AZ
United States
ddatta4@asu.edu

Manasa Pola
Arizona State University
Tempe AZ
United States
mpola@asu.edu

Sam Sunder Golla
Arizona State University
Tempe AZ
United States
sgolla2@asu.edu

Shuddhatm Jain
Arizona State University
Tempe AZ
United States
shuddhatm@asu.edu

ABSTRACT

In the past decade, Object Oriented programming has become one of the most sought-after programming styles in the industry. What makes object oriented programming so appealing is the support for the concepts of Abstraction, Encapsulation, Polymorphism, Inheritance[1], Association, Aggregation and Composition, which make coding efficient, simpler and faster. We present an Augmented Reality Application developed using Unity that aims to teach users the concept of Inheritance[1] by presenting them a user-interactive visual environment. We aim to appeal to the younger audience through this car game because we think inculcating the interest in coding at a young age would not only help the users get a good grasp over the concept but also make the next generation more aware about the precepts of Object Oriented Programming [3]. Our results confirm that users learn the concept of Inheritance[1] faster than they would have learnt in a school environment partly because our application transforms the concept into a visual environment and gives the user the option to choose what is best for them through trial and error.

Please follow the following link to see a demo of our application: https://www.youtube.com/watch?v=qKE1HJjEIqg

KEYWORDS AND PHRASES

Inheritance[1], Open User Model [2], Augmented Reality, Object Oriented Programming [3], Unity [4], Vuforia[5].

1 INTRODUCTION

In layman terms, Inheritance[1] basically means reusability. It is a mechanism in which one class can inherit the features of another class. The features can be fields or methods. The class whose features are inherited is known as a super class or a base class and the class that inherits the above is known as a sub class or a derived class. The very essence of this system is to provide intelligent feedback to the user and for this to suffice, we implemented open user model in our system by incorporating transparency, controllability and visualization.

When the concept of augmented reality was initially introduced in 1992, there was a lot of buzz around its possible applications in the fields of education, management of the information flood, knowledge sharing and organizing distant meetings. However, due to the lack of publicity and the perception of it being 'costly' and a 'luxurious' piece of technology, the buzz eventually faded away. Added to that, the requirement of an additional piece of equipment to render the environment also deterred users from accepting it easily. However, since the beginning of the new century, the buzz has resurfaced, and efforts are being made to justify this buzz with a lot of AR based application starting to make rounds. Games like *Pokémon Go*[6] have helped make augmented reality more mainstream and the use of augmented reality is being explored in various other fields as well.

In the field of adaptive educational systems, Open User Models [2] are a relatively popular concept because their use brings along a number of benefits. A point to be noted however is that, models used in the field of educational systems are different from user models used in personalized information access. Typically, user models used for adaptive educational systems track user knowledge in stark contrast to user interests in models for adaptive information access, and have a relatively distinct structure and the latter models are typically referred to as user profiles.

Augmented Reality renders many uses in diverse domains. One such instance could be educating a user on a technical topic such as Inheritance[1]. In our work, we attempt to do exactly this using an Open User Model [2]. We gave users the ability to make user profiles for personalized evaluation of the progress and expected that this would help users to get motivated when they get additional live feedback from the system on their current performance. This report presents the intuition, workflow and the results of our work. The next section introduces our motivation behind our application, *Inheritance*. The remaining parts of the report describe our design artifacts, methodology, future work and the analysis of the user performance while playing the game, *Inheritance*.

2 MOTIVATIONS

This is the age of smartphones and mobile apps and the younger generation is the primary audience. After practically understanding the building blocks of Object Oriented Programming [3] through years of programming, we realized that the concepts themselves are easy to understand and are very relatable to a lot of common day to day activities. Keeping these things in mind, we realized that we should build an application that would teach a concept of Object Oriented Programming [3] by presenting it in a way that would draw the user's attention and make learning it fun to learn rather than a burden. Since our primary aim was to appeal to the younger audience, we did that by giving the user a car to navigate through an environment containing obstacles, interacting with which, the users will discover new concepts of inheritance. Our aim, as already described in the introduction is to not only inculcate an interest in coding in the users but also to make the next generation more aware about the precepts of Object Oriented Programming [3].

Since we wanted to be innovative in the way a concept is taught to a user, we drifted from the conventional way a concept is taught by incorporating Augmented Reality and used Gamification. We wanted to make the environment adaptive based on users' performance so as to aid the user to learn the concept better.

Nowadays, a lot of emphasis is given to building a far-reaching user model by giving the "adaptation impact" and consolidating components of personalization into the learning model. The essential goal behind any adaptive framework is to gather data from different sources which depend on the user's communication with the framework.

3 DATA ANALYSIS:

In this section we present how and what data we collected from the application and how we used the elements of open user model using the data.

3.1 User Analysis

In order to present user specific statistics, we made it mandatory for every user to login/create a profile so that we can implement user profiling. We tracked all user logins for every user and logged the amount of time a user spent on all scene of the game environment. We also tracked the number of successful hits a user made in every level, which we treat as a good indicator of how much he/she has learnt about inheritance.

3.2 Group Collaboration[9]:

Group collaboration[9] occurs where the individuals involved learn from the activities of the group rather than from another peer. We implemented group collaboration by comparing the user's score to the average score of all users who have played the game and then displaying live on the game on where the user ranks.

3.3 Data Collection:

We logged the user's activities by storing values using SQLite[7] tables. We collected data like total time spent on the game, no of times the user logged in and how much time the user takes for completing each level every time user plays.

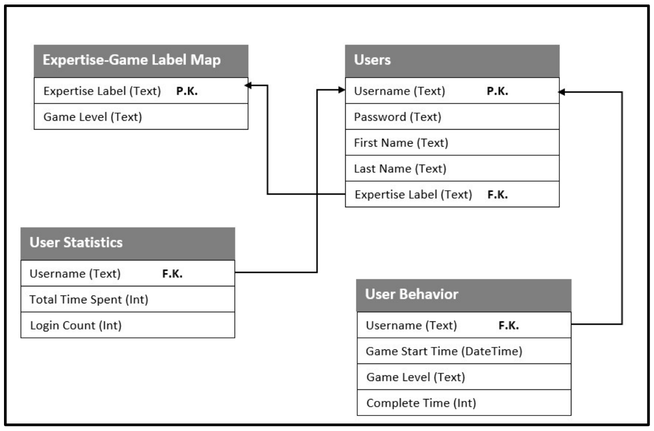


Fig 1. *Inheritance*'s Database Model

The following are the description of some of the main tables that were used:

3.3.1. User Stat Data: This table was used to log data that pertained to the amount of time the user spent using the game. Figure 2 shows the structure and example data that would be logged in the table.

3.3.2. User Behavior: This table was used to log the user performance statistics for every level he/she played on the game. Figure 3 shows the structure and example data that would be logged in the table.

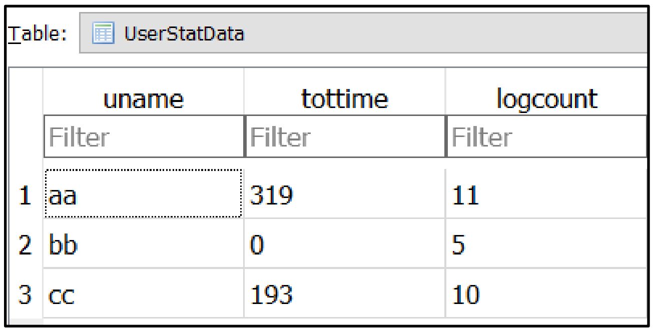


Table: UserStatData

	uname	tottime	logcount
	Filter	Filter	Filter
1	aa	319	11
2	bb	0	5
3	cc	193	10

Fig 2. The *UserStatData* table which logs the username (uname), total time spent on the game (tottime) and the number of times the user logged out (logcount)

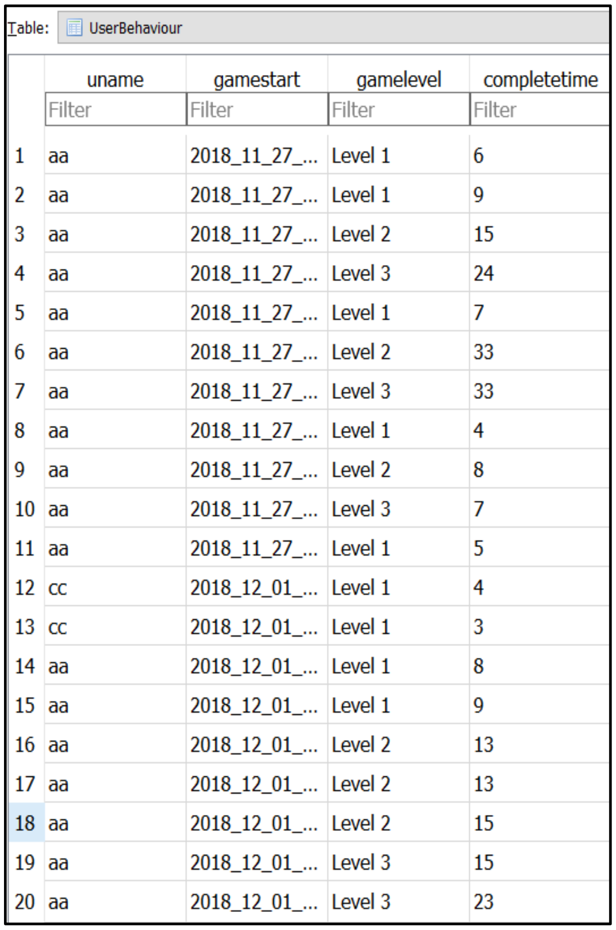


Table: UserBehaviour

	uname	gamestart	gamelevel	completetime
	Filter	Filter	Filter	Filter
1	aa	2018_11_27_...	Level 1	6
2	aa	2018_11_27_...	Level 1	9
3	aa	2018_11_27_...	Level 2	15
4	aa	2018_11_27_...	Level 3	24
5	aa	2018_11_27_...	Level 1	7
6	aa	2018_11_27_...	Level 2	33
7	aa	2018_11_27_...	Level 3	33
8	aa	2018_11_27_...	Level 1	4
9	aa	2018_11_27_...	Level 2	8
10	aa	2018_11_27_...	Level 3	7
11	aa	2018_11_27_...	Level 1	5
12	cc	2018_12_01_...	Level 1	4
13	cc	2018_12_01_...	Level 1	3
14	aa	2018_12_01_...	Level 1	8
15	aa	2018_12_01_...	Level 1	9
16	aa	2018_12_01_...	Level 2	13
17	aa	2018_12_01_...	Level 2	13
18	aa	2018_12_01_...	Level 2	15
19	aa	2018_12_01_...	Level 3	15
20	aa	2018_12_01_...	Level 3	23

Fig 3. The *UserBehaviour* table which logs the username (uname), level start time (gamestart), level number (game level) and the time taken to complete the level (complete time)

3.4 Open User Model[2]

As already described above, Open User Model [2] is a relatively popular concept because of its benefits. User models used for adaptive educational systems usually track user knowledge. Open User Model [2] stands on the pillars of transparency, controllability and visualization and we have implemented then in our application in the following way:

3.4.1 Transparency: We implemented Transparency by comparing the current user's score to the average score of all users and visualized the result LIVE while the user played the game. If no user has played the game before (cold start), the expertise label is set as "*No Previous Record*".

- If the user completes a level in less than the average time taken by all users, the label will show *"Above Average"*.
- If the user completes in more than the average time, the label will show *"Below Average"*.
- If the user completes in the average time, the label will show *"Average"*.

Figure 4 shows an example.

.

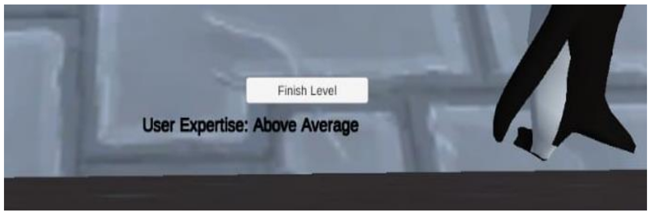


Fig 4. The *User Expertise* label shows the expertise level of the user and is updated live while the user is playing the game

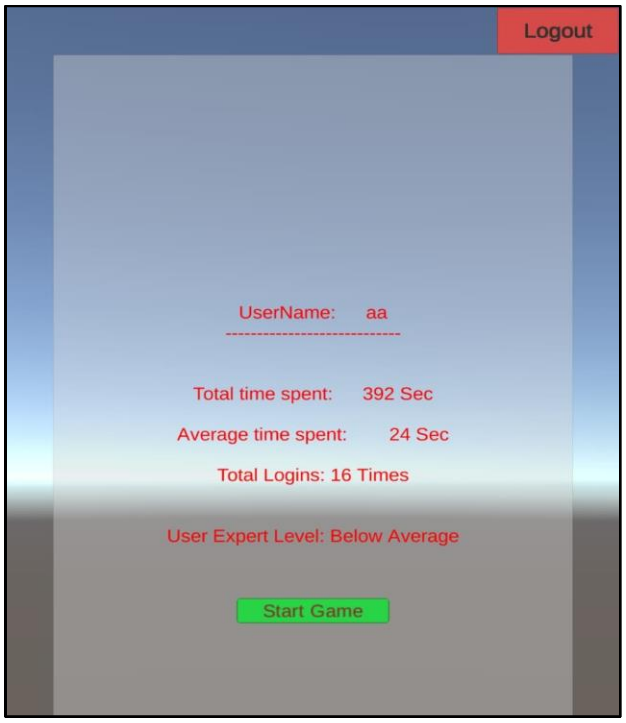
3.4.2 Controllability: We implemented controllability by letting the giving the users control over:

- The objects that the user interacts with while playing the game which is consequential to the user's completion of the level. The user can decide to select the appropriate objects and choose to submit high scores for that level (or vice versa)
- The time the user takes to finish the level is recorded but the user can choose the correct objects to finish faster and choose to submit high scores for that level (or vice versa)

3.4.3 Visualization: We implemented data visualization by visualizing the following aspects of the user log data:

- Username: The name that the user chooses for himself/herself
- Total Time Spent: The total amount of time the user spent playing the game
- Total Logins: The number of times the user has logged into the game
- Average time spent: The average time spent by the user for each time he/she logged in (obtained by dividing total time spent by the number of total logins)
- User Expert Level: The relative rank of the user compared to all other users who have played the game. (Set to 'No Previous Record' if no user has played - to avoid cold start)

This page is shown to the user as soon as he/she logs into the game. Figure 5 shows the page.



UserName: aa

Total time spent: 392 Sec

Average time spent: 24 Sec

Total Logins: 16 Times

User Expert Level: Below Average

Fig 5. The *Data Visualization* page shows the user his/her progress by showing user specific data.

4 METHODOLOGY

The goal of this project is to teach users the concept of Inheritance using Augmented Reality. To help the user learn in an innovative way, we implemented an Open User Model [2].

In our game, a user can register himself and then login with those credentials. After logging in, he/she is taken to the scene on which he can control a 3D object car. The application takes the user through a series of 3 levels in all of which, the task for the user is to collide with other similar objects:

- **Level 1:** This level consists of 3 cars (one of which is the user-controlled car - red in color). Upon colliding with the other cars, the user's car inherits the target's color. The intuition behind this action is to teach users the concept of a child (the red car) inheriting properties (in our case - the color) of a base class (the other cars). The user completes the level by colliding with all objects in the game area. Figure 6 shows an instance of Level 1 being played by a user.

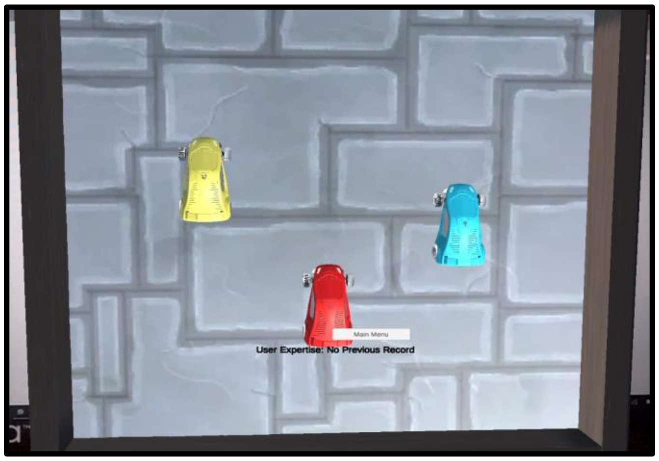


Fig 6. Level 1: The user's car (red in color) has 2 other cars (blue and yellow colored cars) in its game environment and can inherit the color of those cars by colliding with them.

- **Level 2:** The user reaches this level upon completing level 1. This level consists of the user-controlled car (red in color) along with 6 other similar and dissimilar objects in the game area. The six objects are - a white cat, a piece of log, a zombie, a large boulder, a green colored car and a pink colored car. The user has to choose which objects are similar (in our case, the two cars) and collide with them. If the user collides with the other dissimilar objects, it won't do anything consequential to the completion of the level. Similar to level 1, upon colliding with the other cars, the user's car inherits the target's color. The intuition behind this action is to teach users a sub concept of Inheritance[1] that it is always better to inherit from like classes. Figure 7 shows an instance of Level 2 being played by a user.

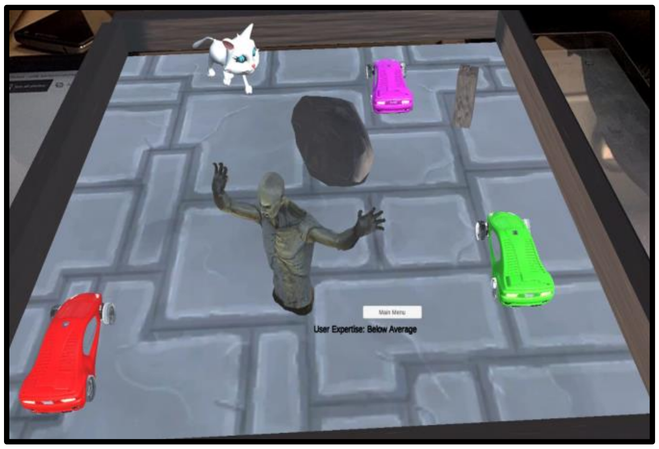


Fig 7. Level 2: The user's car (red in color) has 6 other objects in its game environment. The user must navigate through these objects and collide with only the similar objects.

- **Level 3:** The user reaches this level upon completing level 2. This level has the following characters: user-controlled car (red in color), a cheetah, a white cat and a penguin. Since all objects are dissimilar to a the user's object (the car which is a mechanical object), the user has to choose which object has properties that could be inherited which would enhance the performance of the car. A penguin has the properties of moving slow (as they move by walking). A cat has 4 legs and can run considerably faster than a penguin. Cheetahs are known to be the fastest land animal in the world. We assume that all this information about the properties of the objects is general knowledge and that all users playing our game would have this prerequisite knowledge. The user would then navigate the car and collide with each of the object doing which there will be a considerable change in the speed of the movement of the car. (Cheetah - car moves faster than normal, cat car moves slower than normal, penguin - car moves even slower). In a sense, the car is inheriting the property of speed from the objects. The intuition behind this action is to teach users a sub concept of Inheritance[1] that in an environment where there are no similar base classes having seemingly different properties, it is essential to choose one that would enhance the performance the application. Figure 8 shows an instance of Level 3 being played by a user.

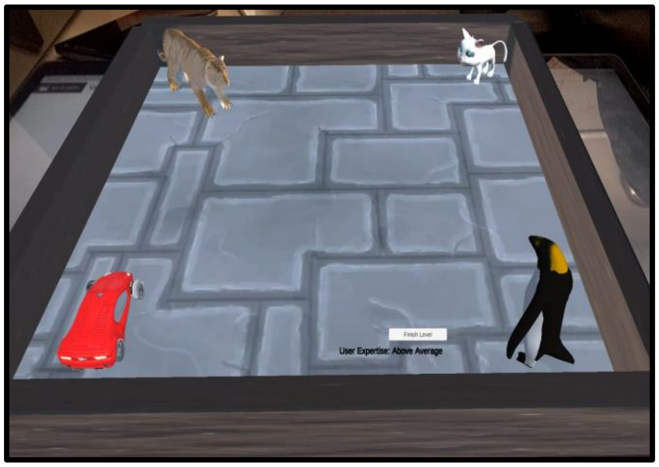


Fig 8. Level 3: The user's car (red in color) has 3 other objects in its game environment. The user has to navigate through these objects collide with an object that would enhance its performance.

Open User Model[2] is implemented to help the user to understand his performance analytics, and thus helps him learn the concept and game better. Using this model, in a sense, we are making the system adaptive to the user. We collected multiple implicit and explicit feedbacks of the user and designed the open user model in such a way to make it adaptive. The implicit feedback collected is the amount of time a user spends on a scene to finish it. Added to that, we also give an option to the user to change the layout and reduce the number of objects/barriers.

The user's object obtaining the colors and speeds of the target objects can be considered as inheritance. It is like reusing the properties of a base class to complete some task. The amount of time spent in this scene can be used to gauge the change in expertise level of the user since the first scene.

5 SYSTEM OVERVIEW

Our application was designed using Unity[4] and used C#[11] as the programming language. For augmented reality we used Vuforia Augmented Reality SDK[5]. For storing data, we used SQLite[7] and exported the application as an android[12] application using the android SDK[12].

The system/application is designed in such a way that the cognitive load on a user is minimal when he/she is using the system. The concept of Gamification is implemented in this system. Thus, a user can apply the typical elements of game playing to learn a technical concept such as Inheritance.
Figure 9 shows a broad perspective of the workflow that our application follows.

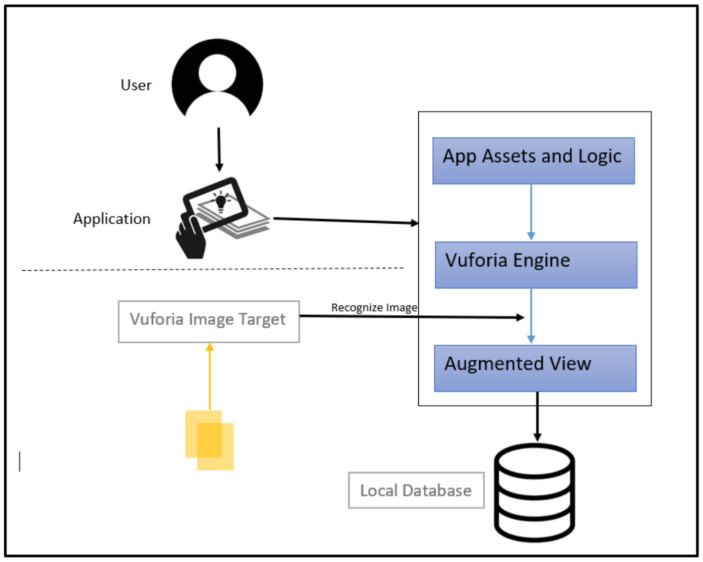


Fig 9. *Workflow*: The user interacts with the application, which sends the interactions to the game engine. The game engine, gets the objects that have to be rendered to the Vuforia engine which renders a 3D image of them on the game environment as soon as it recognizes the image target. The user's interactions are thrown into action on the augmented object and the results are sent over to the local database stored on the hand-held device for post-game applications like user modelling and analytics. Upon demand, the data is picked from the database and sent to the application to be rendered on screen (when the user wants to see his/her stats)

6 OBSERVATIONS

The application captures multiple user data to observe and adapt.

1. We are capturing time spent by user on each level of game.
2. We are analyzing how the user is performing i.e. number of hits to unwanted objects by user, gestures input, to calculate the expertise of user and adapt to the users' skills.
3. One of the future plan is to include how user is interacting with the system after comparing the score with other users.

We use open social model to motivate user by letting him/her compare with other expert user or user at same level and number of attempts made to improve the level of user.
The system is password protected, thus making user feel secure about sharing data.

7 EVALUATION PLAN

We can evaluate the system based on the following things[8],

- Visibility of user analytics: The amount of time spent in each scene, the type of the user.
- Consistency and Standards: We maintained consistency by keeping the scene environment the same.
- Robust: By implementing Open User Model, we made our system Robust.
- Flexibility & Efficiency of use: Novice Users can refer to documentation in case they need to do it.

8 CHALLENGES

These were some of the challenges that we faced before & after development:

1. **Runtime Debugging**: In the Unity platform, we couldn't run the full system in some simulator that will run the code in real android like environment while we get the Debug.Log() statements in the Unity platform. For debug purpose we had to create a dummy function that will put the required log statements in a dummy panel during testing in actual android platform. This part of the code was not required at all, so in final version that is removed.
2. **Free Unity Assets**: The number of free Unity Assets are very limited. For some elements we had to compromise by using the base versions while most elements had to be written from scratch.
3. **Adaptive Navigation Support**: Another important challenge was to design the application in such a way so that there isn't a lot of cognitive load on the user. We avoided this problem by keeping a simple motive for each level that were very intuitive and providing live feedback to the user of his/her performance.

9 DISCUSSIONS & FUTURE WORK

- The application can be used as a beta version and with a set of targeted audience as user we can get their feedback and improve the functionalities based on that.
- Adding more levels in this application by increasing the complexity.
- We can include features to teach various java concepts in this application.
- We can include audio feedback to user for every action user does, thus giving a real time feedback while user is performing the action.
- Data synchronization with server to capture user data globally and using a cloud-based analytics tools.

10 CONCLUSION

As the project is targeted towards learning of Object Oriented Programming [3] concepts for young audience, it is indeed a good idea to help them visualize these concepts in computer and simulate like a real world. Thus, making an augmented reality to provide personalized experience. In this we are proposing a mobile application which helps the user learn Inheritance in a fun and interactive manner. Doing so we can convey the concept visually and learning from these how we can adapt the system to serve the user more fluently. Though with the help of this application we are just addressing a very small concept of Object Oriented Programming Language concept, this opens up a path to create an application at a large-scale application as a tool for all the educational institutions, thus making a big step towards interactive learning.

11 ACKNOWLEDGEMENTS

We would like to extend our sincerest gratitude to Dr. Sharon Hsiao for her support and direction all through the project. We truly value the helpful feedback given to us during the design and development phase of our application. Her insightful remarks helped us improve our application.
We are also thankful to Yihan Lu for giving insightful ideas about the project and code walkthrough.

REFERENCES

[1] Inheritance (Object-oriented Programming)
https://en.wikipedia.org/wiki/inheritance_(object-oriented_programming)
[2] Open User Profiles For Adaptive News Systems: Help Or Harm?
http://wwwconference.org/www2007/papers/paper602.pdf
[3] Object-oriented Programming:
https://en.wikipedia.org/wiki/object-oriented_programming
[4] Unity - Game Engine
https://en.wikipedia.org/wiki/unity_(game_engine)
[5] Vuforia: https://www.vuforia.com/
https://en.wikipedia.org/wiki/Vuforia_Augmented_Reality_SDK
[6] Pokémon Go
https://en.wikipedia.org/wiki/pok%c3%a9mon_go
[7] SQLite
https://en.wikipedia.org/wiki/sqlite
[8]http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6913071&isnumber=6913053

[9] Peer-to-peer And Group Collaboration - Do They Always Match?
https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1376817
[10] Unity Documentation
https://docs.unity3d.com/
[11] C Sharp (programming language)
https://en.wikipedia.org/wiki/C_Sharp_(programming_language)
[12] Android Software Development
https://en.wikipedia.org/wiki/Android_software_development
[14] C# Programming
https://www.tutorialspoint.com/csharp/
[15] StackOverflow for Unity and C# Programming Help
https://stackoverflow.com/