ECE1779 Project A3 Report

Hao Yin - 1007045001

Zhihao Yang - 1002270776

Leo Li - 1002598833

Description

Our web application is a gobang game, also known as “five in a row”. Users can play gobang games with AI as many times as possible. Users can also see game history and replay the last game he or she had with AI. The intelligence level of AI is moderate so this game is suitable for any new player or intermediate player.
The web application is deployed using Lambda and DynamoDB therefore it is capable of scaling automatically according to the usage. What is more, game history in DynamoDB that has expired will be discarded automatically.

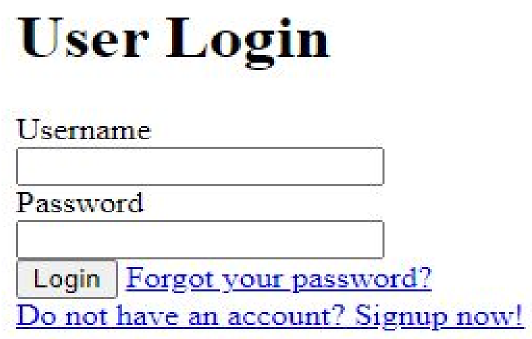
Instructions

User Login Page

To Use our application, simply click the IP address of our Lambda main application:

https://2tbhg94439.execute-api.us-east-1.amazonaws.com/dev

By clicking the link above, you will be directed to the login page of our web application:


User Login

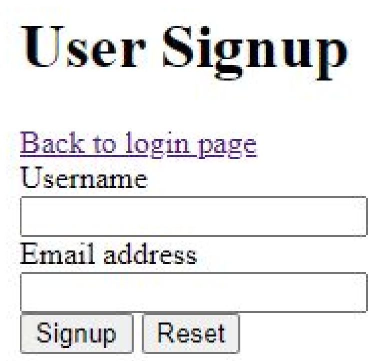
Username

Password

Login Forgot your password?

Do not have an account? Signup now!

You need to sign up if you do not have an account, you do need an email address to complete the signup process:

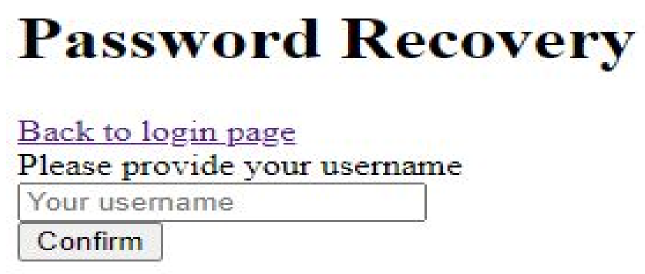

User Signup

Back to login page

Username

Email address

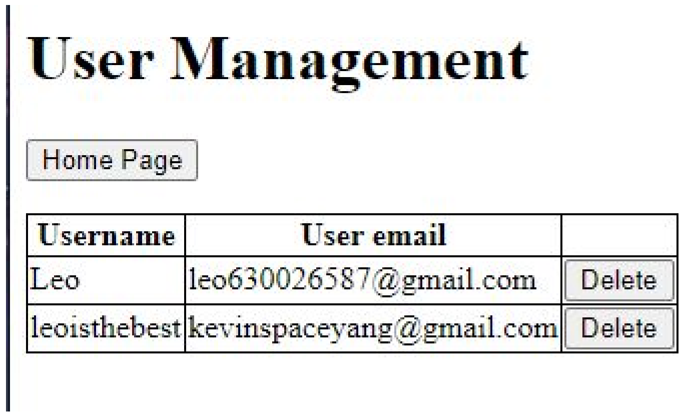
If you forget your password, you can simply click ’Forgot your password’ and follow the instruction to recover your password.


Password Recovery

Back to login page

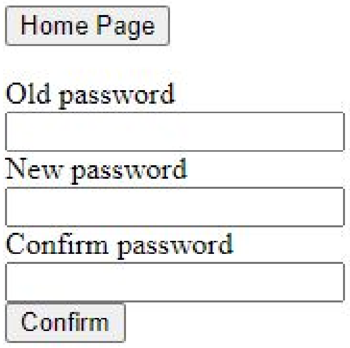
Please provide your username

User Management page:


User Management

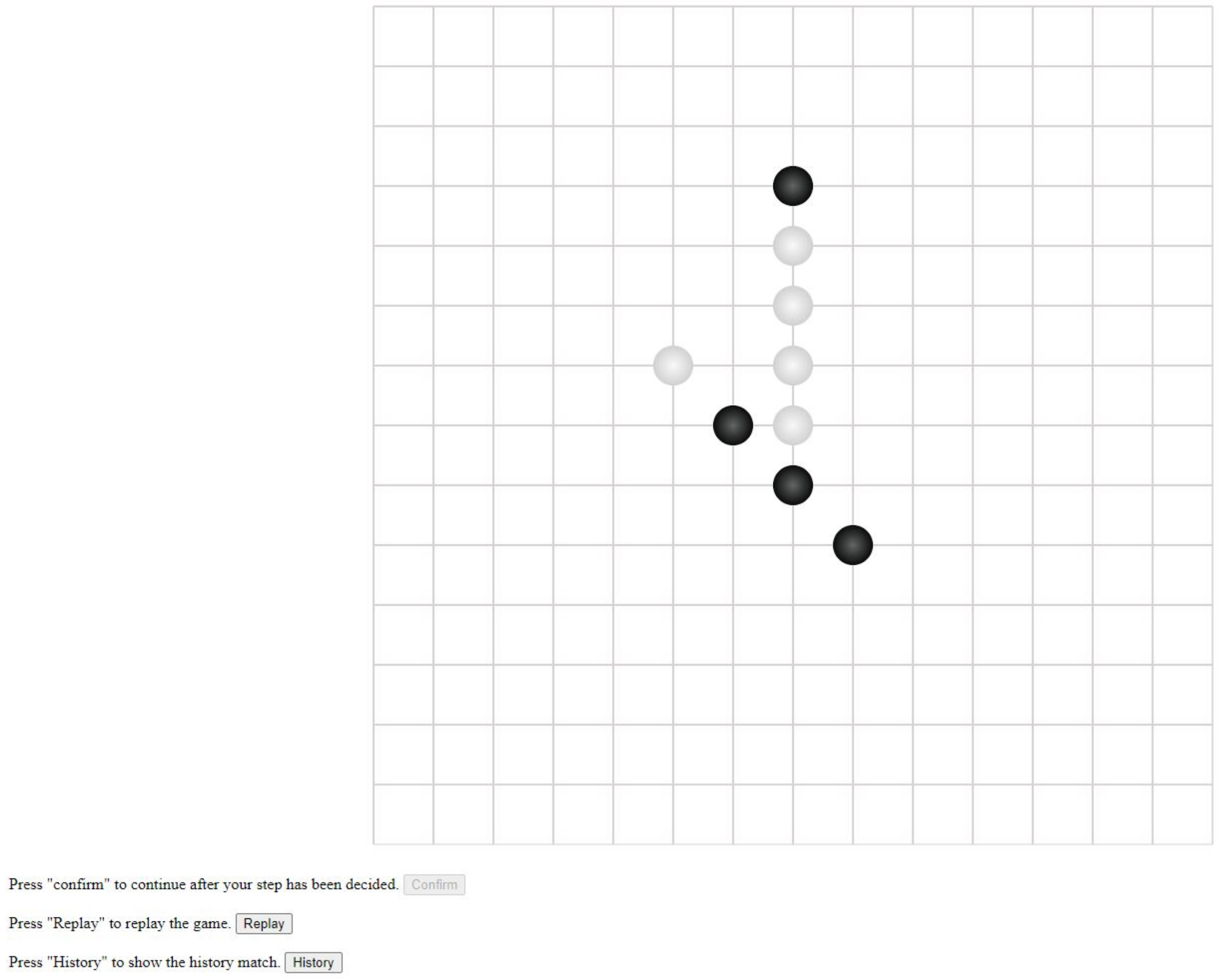
Username	User email	
Leo	leo630026587@gmail.com	Delete
leoisthebest	kevinspaceyang@gmail.com	Delete

You can view all the existing users here, and you are able to delete the existing user, or create a new one. The “Create a new user” button will navigate you to the user registration page.



Once you complete the signup process, you can login with the username and password, if you you login successfully, you can see the home page of our gobang game:

If you want to start playing, click ‘Chess Game’ then you can play with AI. By default you will place a stone first because you are white. Therefore click a spot where you want to place a stone and then click confirm. Each time you have placed a stone, the AI will place a stone accordingly. The one who gets 5 stones in a row wins the game.



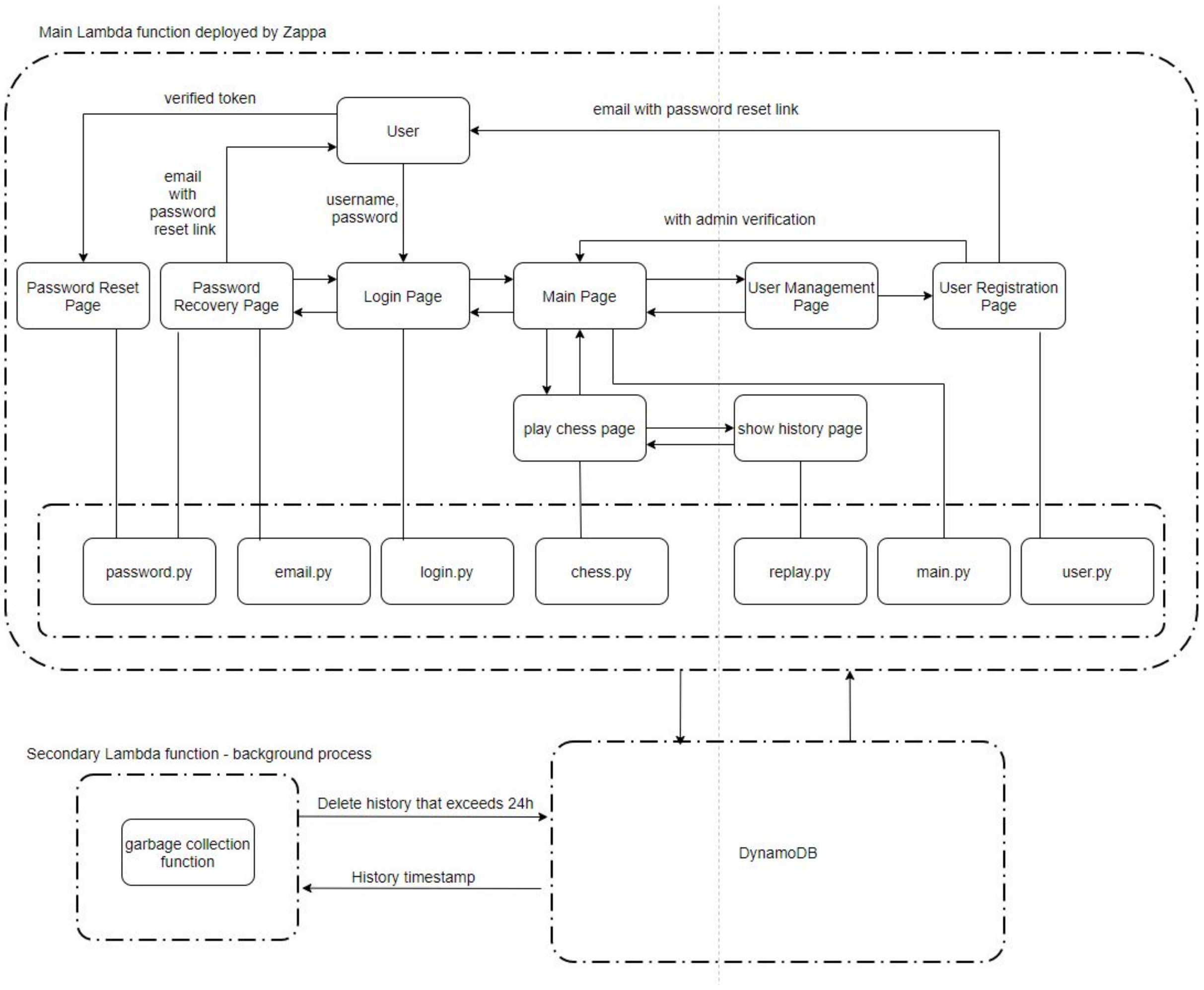
Press "confirm" to continue after your step has been decided.

Press "Replay" to replay the game.

Press "History" to show the history match.

If you would like to start over, press ‘Replay’.
You can also see the game history by pressing ‘History’, then you can browse all the history games that you had with AI. Pressing any game listed in history allows you to replay the game.

System architecture



As shown above, in addition to the Lambda main function, we have a secondary Lambda function, acts as a background process, it checks the all history from DynamoDB periodically and can delete the history that has expired. It does not interact with the Main Lambda function, only interacts with DynamoDB directly and performs garbage collection. By doing this we can save a lot of storage in DynamoDB.

Database schema

Table user_info:

```
{
    "Table": {
        "AttributeDefinitions": [
            {
                "AttributeName": "username",
                "AttributeType": "S"
            },
            {
                "AttributeName": "user_email",
                "AttributeType": "S"
            },
            {
                "AttributeName": "user_password",
                "AttributeType": "S"
            },
            {
                "AttributeName": "user_role",
                "AttributeType": "N"
            }
        ],
        "TableName": "user_info",
        "KeySchema": [
            {
                "AttributeName": "username",
                "KeyType": "HASH"
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableArn": "arn:aws:dynamodb:us-east-1:859113854900:table/user_info",
        "TableId": "4574c830-7f4c-4d21-a03d-c84ebd4a1716"
    }
}
```

Table game_history:

```
{
    "Table": {
        "AttributeDefinitions": [
            {
                "AttributeName": "timestamp",
                "AttributeType": "N"
            },
            {
                "AttributeName": "username",
                "AttributeType": "S"
            },
            {
                "AttributeName": "steps",
                "AttributeType": "S"
            }
        ],
        "TableName": "game_history",
        "KeySchema": [
            {
                "AttributeName": "username",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "timestamp",
                "KeyType": "RANGE"
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableArn": "arn:aws:dynamodb:us-east-1:859113854900:table/game_history",
        "TableId": "20fc0b7f-d6bf-4af5-bacd-57045a97bd2a"
    }
}
```

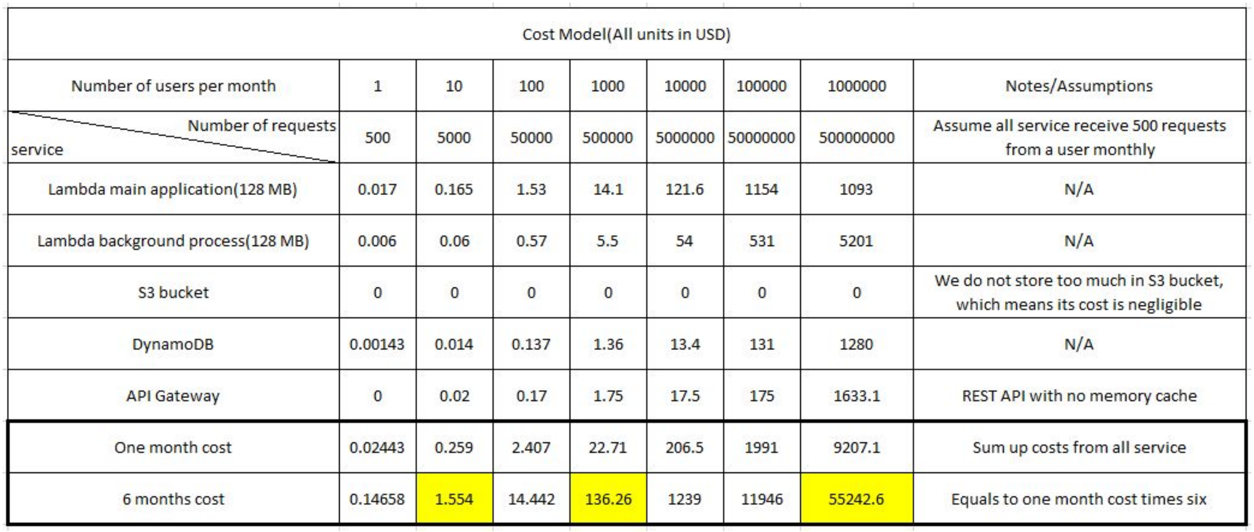
Cost model

Service used:
Lambda main application (estimated without free tier)
Lambda background process (estimated without free tier)
S3 bucket (does not exceed free tier so negligible)
DynamoDB
API Gateway

Assumptions:
All service receive 500 requests from a user monthly
For more assumptions please refer to the table below

Estimate Tool:
AWS Pricing Calculator: https://calculator.aws/#/
Note: All estimations below are calculated using this estimate tool.

Cost Model:



Cost Model(All units in USD)								
Number of users per month	1	10	100	1000	10000	100000	1000000	Notes/Assumptions
Number of requests / service	500	5000	50000	500000	5000000	50000000	500000000	Assume all service receive 500 requests from a user monthly
Lambda main application(128 MB)	0.017	0.165	1.53	14.1	121.6	1154	1093	N/A
Lambda background process(128 MB)	0.006	0.06	0.57	5.5	54	531	5201	N/A
S3 bucket	0	0	0	0	0	0	0	We do not store too much in S3 bucket, which means its cost is negligible
DynamoDB	0.00143	0.014	0.137	1.36	13.4	131	1280	N/A
API Gateway	0	0.02	0.17	1.75	17.5	175	1633.1	REST API with no memory cache
One month cost	0.02443	0.259	2.407	22.71	206.5	1991	9207.1	Sum up costs from all service
6 months cost	0.14658	1.554	14.442	136.26	1239	11946	55242.6	Equals to one month cost times six

Discussion:
According to the estimation above, we can conclude that if we have 1000000 active users for 6 months, the estimated cost would be 55k USD. This estimation is based on 500 requests from a user monthly. If they are heavy users, say, 500 requests per day, the cost will dramatically increase.

Contributing

Gobang AI algorithm with Python:

https://blog.csdn.net/marble_xu/article/details/90450436?ops_request_misc=%25257B%252522request%25255Fid%252522%25253A%252522160756383219195265193999%252522%25252C%252522scm%252522%25253A%25252220140713.130102334..%252522%25257D&request_id=160756383219195265193999&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~blog~top_click~default-2-90450436.pc_v1_rank_blog_v1&utm_term=python%20%E4%BA%94%E5%AD%90%E6%A3%8BAI%E5%AE%9E%E7%8E%B0

Contact

leoo.li@mail.utoronto.ca
hy.yin@mail.utoronto.ca
kevinspace.yang@mail.utoronto.ca