

| | |
|--|----------|
| Models ----- | 2 |
| User:..... | 2 |
| Event: | 2 |
| Goal:..... | 3 |
| TimeRestriction: | 3 |
| Views ----- | 4 |
| AddUser: | 4 |
| EditUser: | 4 |
| GetUser: | 5 |
| DeleteUser: | 5 |
| GetAllUsers: | 6 |
| AddEvent: | 7 |
| EditEventForUser:..... | 7 |
| GetEventsInTimeRange: | 8 |
| GetEventsInTimeRangeForUser:..... | 9 |
| GetAllEvents: | 10 |
| DeleteEventForUser: | 11 |
| AddGoal:..... | 11 |
| EditGoalById:..... | 11 |
| DeleteGoalForUser: | 11 |
| GetGoalById:..... | 12 |
| GetAllCompletedGoalsForUser: | 12 |
| GetAllIncompletedGoalsForUser:..... | 13 |
| AddTimeRestriction: | 13 |
| EditTimeRestrictionForUser: | 14 |
| DeleteTimeRestrictionForUser: | 14 |
| GetTimeRestrictionForUser: | 14 |
| GetTimeRestrictionsInTimeRangeForUser: | 15 |
| GetTimeRestrictionsWithStartAndEndForUser: | 16 |
| GetTimeRestrictionsWithFrequencyForUser:..... | 17 |
| GetAllTimeRestrictionsForUser:..... | 18 |

Early Bird v0.2.0 API

Models

All model logic is located in Group-G-Fall-2019/API/EarlyBirdApi/Api/models.py.

User:

The User model represents an account for the Early Bird Android app. Its JSON representation looks like:

```
{
  "Id": int,
  "Username": string,
  "Email": string,
  "Passwd": hashed string,
  "Gender": string,
  "FirstName": string,
  "LastName": string
}
```

Event:

The Event model represents an event that is scheduled for the User. Its JSON representation looks like:

```
{
  "Id": int,
  "Name": string,
  "StartDate": string,
  "EndDate": string,
  "NotificationDate": string,
  "Frequency": string,
  "UserId": int
}
```

Goal:

The Goal model represents a goal or reminder that the User wants to achieve. Its JSON representations looks like:

```
{
  "Id": int,
  "Name": string,
  "IsCompleted": bool,
  "Notes": string,
  "UserId": int
}
```

TimeRestriction:

The TimeRestriction model represents a range of time where the User does not want anything scheduled. Its JSON representation looks like:

```
{
  "Id": int,
  "StartDate": string,
  "EndDate": string,
  "Frequency": string,
  "UserId": int
}
```

Views

All view logic is located in Group-G-Fall-2019/API/EarlyBirdApi/Api/views.py.

AddUser:

URL Path: http://127.0.0.1:8000/AddUser/

HTTP Method: POST

Returns: HTTP Status

Parameters: JSON object in the form of

```
{
  "Username": "username",
  "Email": "email@mail.com",
  "Passwd": "hashedSequenceOfCharacters",
  "Gender": "Woman",
  "FirstName": "Sally",
  "LastName": "Johnson"
}
```

Notes: The Id property IS NOT sent with the JSON object. The Id property is automatically generated when entering the database.

EditUser:

URL Path: http://127.0.0.1:8000/EditUser/username/property/newData/

HTTP Method: PATCH

Returns: HTTP Status

Parameters: username - string, property - string, newData - string

Notes: Cannot update the Id property or Username property.

GetUser:

URL Path: http://127.0.0.1:8000/GetUser/username/

HTTP Method: GET

Returns: HTTP Status or a JSON object in the form of

```
{  
  "UserId": 1,  
  "Username": "username",  
  "Email": "email@mail.com",  
  "Passwd": "hashedSequenceOfCharacters",  
  "Gender": "Woman",  
  "FirstName": "Sally",  
  "LastName": "Johnson"  
}
```

Parameters: username - string

DeleteUser:

URL Path: http://127.0.0.1:8000/DeleteUser/username/

HTTP Method: DELETE

Returns: HTTP Status

Parameters: username - string

GetAllUsers:

URL Path: http://127.0.0.1:8000/GetAllUsers/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "Username": "username",
    "Email": "email@mail.com",
    "Passwd": "hashedSequenceOfCharacters",
    "Gender": "Woman",
    "FirstName": "Sally",
    "LastName": "Johnson"
  },
  {
    "Id": 2,
    "Username": "username2",
    "Email": "email2@mail.com",
    "Passwd": "hashedSequenceOfCharacters",
    "Gender": "Transgendered Woman",
    "FirstName": "Joe",
    "LastName": "Thomas"
  }
]
```

Parameters: None

Notes: Will return all Users in the database.

AddEvent:

URL Path: http://127.0.0.1:8000/AddEvent/

HTTP Method: POST

Returns: HTTP Status

Parameters: JSON object in the form of

```
{  
  "Name": "name",  
  "StartDate": "2019-01-01T13:00:00",  
  "EndDate": "2019-01-01T15:00:00",  
  "NotificationDate": "2019-01-01T12:45:00",  
  "Frequency": "Weekly",  
  "UserId": 1  
}
```

Notes: The Id property IS NOT sent with the JSON object. The Id property is automatically generated when entering the database.

EditEventForUser:

URL Path:

http://127.0.0.1:8000/EditEventForUser/username/name/start/end/notification/frequency/
property/newData/

HTTP Method: PATCH

Returns: HTTP Status

Parameters: username - string, start - string, end - string, notification - string, frequency
- string, property - string, newData - string

Notes: Cannot update the Id property or UserId property.

GetEventsInTimeRange:

URL Path: http://127.0.0.1:8000/GetEventsInTimeRange/start/end/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "Name": "name",
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "NotificationDate": "2019-01-01T12:45:00",
    "Frequency": "Weekly",
    "UserId": 1
  },
  {
    "Id": 2,
    "Name": "name",
    "StartDate": "2019-01-02T13:00:00",
    "EndDate": "2019-01-02T15:00:00",
    "NotificationDate": "2019-01-02T12:45:00",
    "Frequency": "Monthly",
    "UserId": 2
  }
]
```

Parameters: start - string, end - string

Notes: This method returns all Events that fall within the time range, regardless of User. This means that the method will most likely return a list of Events with various User owners. IMPORTANT: start is inclusive, end is exclusive.

GetEventsInTimeRangeForUser:

URL Path: http://127.0.0.1:8000/GetEventsInTimeRangeForUser/username/start/end/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "Name": "name",
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "NotificationDate": "2019-01-01T12:45:00",
    "Frequency": "Weekly",
    "UserId": 1
  },
  {
    "Id": 2,
    "Name": "name",
    "StartDate": "2019-01-01T15:30:00",
    "EndDate": "2019-01-01T16:00:00",
    "NotificationDate": "2019-01-01T15:00:00",
    "Frequency": "Once",
    "UserId": 1
  }
]
```

Parameters: username - string, start - string, end - string

Notes: This returns a list of Events that fall within a given time range for a particular User.
IMPORTANT: start is inclusive, end is exclusive.

GetAllEvents:

URL Path: http://127.0.0.1:8000/GetAllEvents/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "Name": "name",
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "NotificationDate": "2019-01-01T12:45:00",
    "Frequency": "Weekly",
    "UserId": 1
  },
  {
    "Id": 2,
    "Name": "name",
    "StartDate": "2019-01-01T15:30:00",
    "EndDate": "2019-01-01T16:00:00",
    "NotificationDate": "2019-01-01T15:00:00",
    "Frequency": "Once",
    "UserId": 1
  },
  {
    "Id": 3,
    "Name": "name",
    "StartDate": "2019-01-02T13:00:00",
    "EndDate": "2019-01-02T15:00:00",
    "NotificationDate": "2019-01-02T12:45:00",
    "Frequency": "Monthly",
    "UserId": 2
  }
]
```

Parameters: None

Notes: Will return all Events in the database.

DeleteEventForUser:

URL Path: http://127.0.0.1:8000/DeleteEventForUser/username/name/start/end/

HTTP Method: DELETE

Returns: HTTP Status

Parameters: username - string, name - string, start - string, end - string

Notes: Deletes a specific Event for a particular User. IMPORTANT: start is inclusive, end is exclusive.

AddGoal:

URL Path: http://127.0.0.1:8000/AddGoal/

HTTP Method: POST

Returns: HTTP Status

Parameters: JSON object in the form of

```
{  
    "Name": "name",  
    "IsCompleted": false,  
    "Notes": "Various User-made notes.",  
    "UserId": 1  
}
```

Notes: The Id property IS NOT sent with the JSON object. The Id property is automatically generated when entering the database.

EditGoalById:

URL Path: http://127.0.0.1:8000/EditGoalById/id/property/newData/

HTTP Method: PATCH

Returns: HTTP Status

Parameters: id - int, property - string, newData - string

Notes: If editing IsCompleted property of the Goal, make sure to enter either "true" or "false" as a string in the newData url parameter.

DeleteGoalForUser:

URL Path: http://127.0.0.1:8000/DeleteGoalForUser/username/name/isCompleted/notes/

HTTP Method: DELETE

Returns: HTTP Status

Parameters: username - string, name - string, isCompleted - string, notes - string

Notes: Deletes a specific Goal for a particular User. IMPORTANT: You MUST input isCompleted as a string and not a boolean. True should be put in as "true". False should be put in as "false". This is due to some limitations of Django URL parameters.

GetGoalById:

URL Path: http://127.0.0.1:8000/GetGoalById/id/

HTTP Method: GET

Returns: HTTP Status or a JSON object in the form of

```
{
  "Id": 1,
  "Name": "name",
  "IsCompleted": true,
  "Notes": "Various User-made notes.",
  "UserId": 1
}
```

Parameters: id - int

GetAllCompletedGoalsForUser:

URL Path: http://127.0.0.1:8000/GetAllCompletedGoalsForUser/username/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "Name": "name",
    "IsCompleted": true,
    "Notes": "Various User-made notes.",
    "UserId": 1
  },
  {
    "Id": 2,
    "Name": "name",
    "IsCompleted": true,
    "Notes": "Various User-made notes.",
    "UserId": 1
  }
]
```

Parameters: username - string

GetAllIncompletedGoalsForUser:

URL Path: http://127.0.0.1:8000/GetAllIncompletedGoalsForUser/username/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "Name": "name",
    "IsCompleted": false,
    "Notes": "Various User-made notes.",
    "UserId": 1
  },
  {
    "Id": 2,
    "Name": "name",
    "IsCompleted": false,
    "Notes": "Various User-made notes.",
    "UserId": 1
  }
]
```

Parameters: username - string

AddTimeRestriction:

URL Path: http://127.0.0.1:8000/AddTimeRestriction/

HTTP Method: POST

Returns: HTTP Status

Parameters: JSON object in the form of

```
{
  "StartDate": "2019-01-01T13:00:00",
  "EndDate": "2019-01-01T15:00:00",
  "Frequency": "Weekly",
  "UserId": 1
}
```

Notes: The Id property IS NOT sent with the JSON object. The Id property is automatically generated when entering the database.

EditTimeRestrictionForUser:

URL Path:

http://127.0.0.1:8000/EditTimeRestrictionForUser/username/start/end/frequency/property/newData/

HTTP Method: PATCH

Returns: HTTP Status

Parameters: username - string, start - string, end - string, frequency - string, property - string, newData - string

DeleteTimeRestrictionForUser:

URL Path:

http://127.0.0.1:8000/DeleteTimeRestrictionForUser/username/start/end/frequency/

HTTP Method: DELETE

Returns: HTTP Status

Parameters: username - string, start - string, end - string, frequency - string

Notes: Deletes a specific TimeRestriction for a particular User. IMPORTANT: start is inclusive, end is exclusive.

GetTimeRestrictionForUser:

URL Path:

http://127.0.0.1:8000/GetTimeRestrictionForUser/username/start/end/frequency/

HTTP Method: GET

Returns: HTTP Status or a JSON object in the form of

```
{  
  "Id": 1,  
  "StartDate": "2019-01-01T13:00:00",  
  "EndDate": "2019-01-01T15:00:00",  
  "Frequency": "Monthly",  
  "UserId": 1  
}
```

Parameters: username - string, start - string, end - string, frequency - string

GetTimeRestrictionsInTimeRangeForUser:

URL Path:

http://127.0.0.1:8000/GetTimeRestrictionsInTimeRangeForUser/username/start/end/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "Frequency": "Monthly",
    "UserId": 1
  },
  {
    "Id": 2,
    "StartDate": "2019-01-01T15:00:00",
    "EndDate": "2019-01-01T16:00:00",
    "Frequency": "Daily",
    "UserId": 1
  }
]
```

Parameters: username - string, start - string, end - string

Notes: Start is inclusive, end is exclusive.

GetTimeRestrictionsWithStartAndEndForUser:

URL Path:

http://127.0.0.1:8000/GetTimeRestrictionsWithStartAndEndForUser/username/start/end/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "Frequency": "Monthly",
    "UserId": 1
  },
  {
    "Id": 2,
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "Frequency": "Daily",
    "UserId": 1
  }
]
```

Parameters: username - string, start - string, end - string

Notes: This method is not using a time range, it is finding TimeRestrictions that have StartDate == start and EndDate == end.

GetTimeRestrictionsWithFrequencyForUser:

URL Path:

http://127.0.0.1:8000/GetTimeRestrictionsWithFrequencyForUser/username/frequency/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "Frequency": "Monthly",
    "UserId": 1
  },
  {
    "Id": 2,
    "StartDate": "2019-01-01T15:00:00",
    "EndDate": "2019-01-01T16:00:00",
    "Frequency": "Monthly",
    "UserId": 1
  }
]
```

Parameters: username - string, frequency - string

GetAllTimeRestrictionsForUser:

URL Path: http://127.0.0.1:8000/GetAllTimeRestrictionsForUser/username/

HTTP Method: GET

Returns: HTTP Status or a list of JSON objects in the form of

```
[
  {
    "Id": 1,
    "StartDate": "2019-01-01T13:00:00",
    "EndDate": "2019-01-01T15:00:00",
    "Frequency": "Monthly",
    "UserId": 1
  },
  {
    "Id": 2,
    "StartDate": "2019-01-01T15:00:00",
    "EndDate": "2019-01-01T16:00:00",
    "Frequency": "Daily",
    "UserId": 1
  }
]
```

Parameters: username - string